RESEARCH
IN LOGISTICS
& PRODUCTION

EXTRACTING PROCESS-RELATED INFORMATION FROM ERP SYSTEMS FOR PROCESS DISCOVERY

Wojciech Fliegner*

* Poznan University of Economics, Poland, Email: wojciech.fliegner@ue.poznan.pl

Abstract This paper discusses various aspects that should be considered when defining and executing extraction process–related information from the source data (ERP system) to an event log. This includes trace and event selection, as well as important project decisions that should be made beforehand. The basic idea of a proposed approach is demonstrated by performing a case study which is related to a standard sales order processing, that is, business process from sales quotation through sales order and delivery to invoicing. For this processes we showed the characteristics of the event logs, and the models we can discover.

Paper type: Research Paper

Published online: 30 October 2014
Vol. 4, No. 4, pp. 315-329

ISSN 2083-4942 (Print)
ISSN 2083-4950 (Online)

Keywords: *process discovery, business process management, ERP systems*

1. INTRODUCTION

Business Process Management (BPM) is the discipline that combines knowledge from information technology and knowledge from management sciences and applies this to operational business processes (van der Aalst, 2004), (Weske, 2007). It has received considerable attention in recent years due to its potential for significantly increasing productivity and saving costs. Moreover, today there is an abundance of BPM systems. These systems are generic software systems that are driven by explicit process designs to enact and manage operational business processes (van der Aalst, ter Hofstede & Weske, 2003).

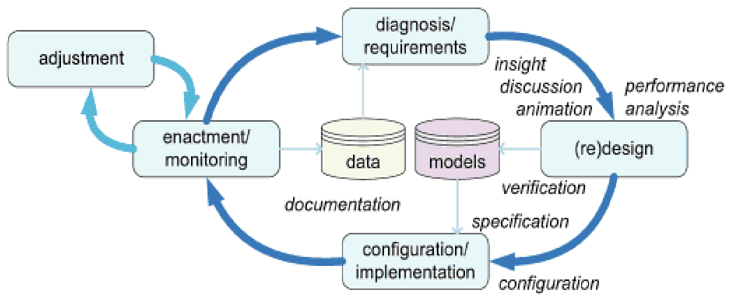

Fig. 1 The BPM life-cycle (adapted from van der Aalst, 2011)

Figure 1 describes the different phases of managing a particular business process. In the design phase, a process is designed. This model[1] is transformed into a running system in the configuration/implementation phase. After the Workflow Management (WFM) or BPM system supports the designed processes, the enactment/monitoring phase starts. In this phase, the processes are running while being monitored by management to see if any changes are needed. Some of these changes are handled in the adjustment phase (in this phase, the process is not redesigned and no new software is created; only predefined controls are used to adapt or reconfigure the process). The diagnosis/requirements phase evaluates the process and monitors emerging requirements due to changes in the environment of the process (e.g., changing policies, laws, competition). Poor performance (e.g., inability to meet service levels) or new demands imposed by the environment may trigger a new iteration of the BPM life-cycle starting with the redesign phase.

As Fig. 1 shows, process models play a dominant role in the (re)design and configuration/implementation phases, whereas data plays a dominant role in the enactment/monitoring and diagnosis/requirements phases. The figure also lists the different ways in which process models are used. Until recently, there were few connections between the data produced while executing the process and the actual process design. In fact, in most organizations the diagnosis/requirements phase is not supported in a systematic and continuous manner. Only severe problems or major external changes will trigger another iteration of the life-cycle, and factual information about the current process is not actively used in redesign decisions. We propose to use process mining techniques[i] to truly "close" the BPM life-cycle. Data recorded by information systems (in audit trails, transaction logs, databases, and so forth) can provide a better view on the actual processes, i.e., deviations can be analyzed and the quality of models can be improved.

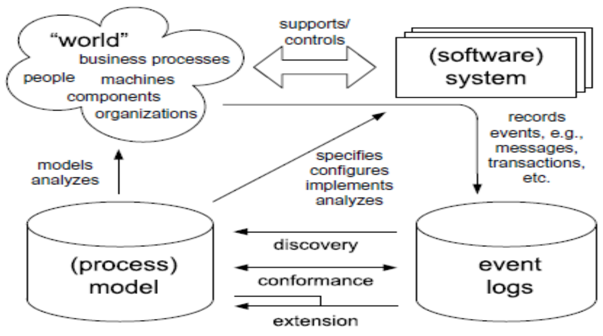

Fig. 2 Positioning of the process mining (Process Mining, 2012)

Figure 2 shows that process mining establishes links between the actual processes and their data on the one hand and process models on the other hand. Today's information systems (such as the traditional WFM systems, ERP (Enterprise Resource Planning) systems, CRM (Customer Relationship Management) systems, rule-based systems, call center software, high-end middleware, etc.) provide detailed information about the activities that have been executed. Figure 2 refers to such data as event logs.

However, most information systems store such information in unstructured form, e.g., event data is scattered over many tables or needs to be tapped off from subsystems exchanging messages. In such cases, event data exist but some efforts are needed to extract them. Data extraction is an integral part of any process mining effort and a main step in performing process discovery on such systems is therefore to properly construct an event log from the logged data. Event logs can be used to conduct three types of process mining as shown in Fig. 2.

The first type of process mining is discovery. A discovery technique takes an event log and produces a process model[2] explaining the behavior recorded in the log[3]. If the event log contains information about resources, one can also (beside the control–flow models) discover resource-related models, e.g., a social network showing how people work together in an organization.

The second type of process mining is conformance. Here, an existing process model is compared with an event log of the same process. Conformance checking can be used to check if reality, as recorded in the log, conforms to the model and vice versa. Conformance checking may be used to detect, locate and explain deviations, and to measure the severity of these deviations.

The third type of process mining is enhancement. Here, the idea is to extend or improve an existing process model using information about the actual process recorded in some event log. Whereas conformance checking measures the alignment between model and reality, this third type of process mining aims at changing or extending the a priori model. One type of enhancement is repair, i.e., modifying the model to better reflect reality. Another type of enhancement is extension, i.e., adding a new perspective to the process model by cross-correlating it with the log.

Orthogonal to the three types of mining (discovery, conformance, and enhancement), following perspectives can be identified:

- Frequent principal divided on subsections,the control-flow perspective focuses on the control-flow, i.e. the ordering of activities; the goal of mining this perspective is to find a good characterization of all possible paths, e.g., expressed in terms of a Petri net or some other notation (e.g., EPCs, BPMN, and UML ADs),
- the organizational perspective focuses on information about resources hidden in the log, i.e., which actors (e.g., people, systems, roles, and departments) are involved and how are they related; the goal is to either structure the organization by classifying people in terms of roles and organizational units or to show the social network,
- the case perspective focuses on properties of cases; obviously, a case can be characterized by its path in the process or by the originators working on it; however, cases can also be characterized by the values of the corresponding data elements,
- the time perspective is concerned with the timing and frequency of events; when events bear timestamps it is possible to discover bottlenecks measure service levels, monitor the utilization of resources, and predict the remaining processing time of running cases.

Note that the different perspectives are partially overlapping and non-exhaustive. Nevertheless, they provide a good characterization of the aspects that process mining aims to analyze.

2. CONTENTS OF EVENT LOGS

Extracting an event log can be regarded as a crucial step in a process discovery project. The structure and contents of an event log determines the view on the process and the process discovery results that can be retrieved.

In general an event log records the events that occur in a certain process for a certain case. Figure 3 visualizes the general structure of event logs and their relation to the definition of a process with activities.

The process definition specifies which activities should be executed and in which structure. When a new case is started a new instance of the process is generated which is called a process instance. This process instance might leave a trace of events that are executed for that case in the event log. An event log consists of any number of traces. Each trace describes a sequential list of events corresponding to a particular case of the logged process.

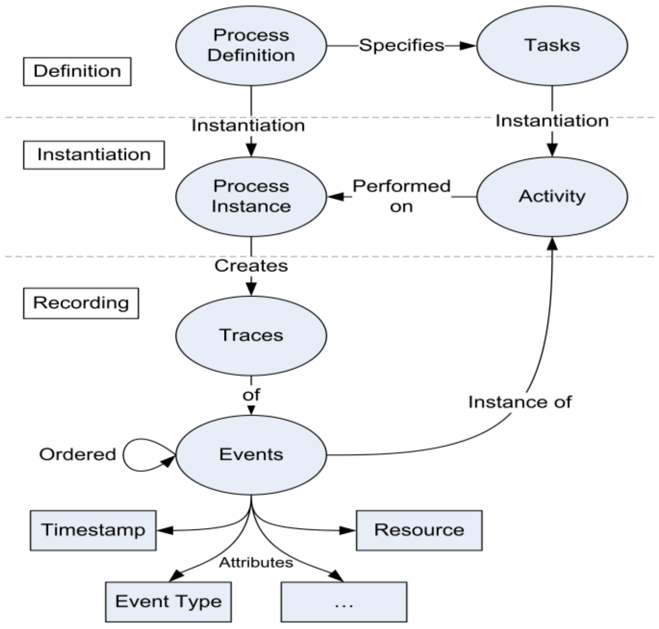

Fig. 3 General event log structure

Each event is an instance of a certain activity as defined within the process definition. Events represent atomic states of an activity[4] that have been observed during the execution of a process and they are ordered to indicate in which sequence activities have occurred. In most cases this order is defined by the date and time or timestamp attribute of the event. Sometimes the start and stop information is recorded of a single activity. This is recorded in the event type attribute of the event. Another common attribute is the resource that executed the event which can be a user of the system, the system itself or an external system. Many other attributes can be stored within the event log related to the event for example the data attributes added or changed in the activity.

Unfortunately, not every information system logs events in the described way. Information about the relation between events and activities or even between traces and process instances is often not recorded. The main reason for this is that the event log is not seen as a recording of the execution of a process by system designers. In some systems the event log is used for debugging errors encountered in the system. In other systems events need to be derived from data scattered over various tables.

3. EXTRACTION FROM THE ERP DATABASES TO DESIRED EVENT LOGS - CHALLENGES

In general the extraction consists of two steps. The first step is the definition of the extraction. This extraction definition specifies how concepts of the data source are mapped onto concepts of the event log. The second step is to execute this conversion and actually convert the data from the data source to the event log as specified in the mapping. It is important to be aware of the influence of decisions made in the mapping phase on the resulting event log and hence the process discovery results. This section discusses several key aspects that are important in defining the extraction.

The extraction should be easy to use for people without a deep programming or process discovery background. Therefore the amount of programming required should be limited. To avoid programming we use the Structured Query Language (SQL)[5].

It is assumed that the data is available in the form of a relational database. Having this data, the first step is to create an SQL query from the extraction definition for each log, trace and event instance[6]. The second step is to run each of these queries on the source system's database. The results of these queries are to be stored in an intermediate database. Before process analysis techniques such as process discovery can be applied on these event logs this intermediate database should be converted in the third step to a standardized format[7]. Extracting event logs may be very challenging. Defining a conversion is often not trivial and several choices should be made. Here, we discuss the six most important challenges.

Challenge 1: goal and scope of the process discovery project

The goal of the process discovery project should be clear before the event log extraction can start. In many cases the executed process is not known and the goal of process discovery therefore is to visualize the executed process as recorded in the event log. Another goal of a process discovery project can for example be to look at certain performance aspects of the process and their possible causes. As a result of the goal of the project the scope can be determined. This is important for the extraction because it determines what should or shouldn't be included in the event log. Enterprise information systems may have thousands of tables with business - relevant data. How to decide which tables to incorporate? Domain knowledge is needed to locate the required data and to scope it. Obviously, the desired scope depends on both the available data and the questions that need to be answered.

Challenge 2: trace selection

It is important to note that in one event log there is only one type of process instance to which the events are related. Candidate process instances are often recognized business objects. Business objects are objects or items handled or used by the business. Examples include patients, machines, robots, orders, invoices and insurance claims. Sometimes trace candidates can also be less tangible objects such as treatments in a hospital, machine usage between start up and shut down, user sessions on a web site, e-mail conversations etc. A trace should contain events related to a single business object. This common business object type is our process instance.

The selection of the trace determines or is determined by the scope of the project. By selecting a certain business object as the trace the scope is set. Consider for example a hospital which treats patients following certain treatment procedures. If one patient is considered to be the process instance one can investigate which treatments patients undergo including possible relationships between different treatments. However, if a treatment itself is taken to be the process instance, as could be defined by the scope of the project, no information about the overall process the patient has gone through is available. In the latter case only separate treatment executions can be investigated and therefore the project scope is smaller than when a patient is considered to be the process instance. One could even select the process instance to be a machine used in several different treatments such as an X-ray machine. These machines often keep detailed records of their usage. When selecting such a machine as a process instance one looses detailed information about the treatment process and the process the patient went through. Another example is a procurement process which includes multiple business objects to follow. Possible candidates are order requests, orders, order lines, receipt of goods or invoices. Each one is in theory a valid choice and cer-tain events could be related to all these objects. Nonetheless each object has a different set of related events. Hence, process discovery results will differ depending on the trace selection.

Challenge 3: snapshots

Cases may have a lifetime extending beyond the recorded period, e.g., a case was started before the beginning of the event log or was still running when the recording stopped. Therefore, it is important to realize that event logs typically just provide a snapshot of a longer running process. When the average duration of a case is short compared to the length of the recording, it is best to solve this problem by removing incomplete cases. In many cases, the initial and final activities are known, thus making it easy to filter the event log: simply remove all cases with a missing “head” or “tail”. However, when the average duration of a case is of the same order of magnitude as the length of the recording, it becomes difficult to discover end to-end processes.

Challenge 4: event selection

After the selection of the process instance to which events should relate, the selection and specification of these events can begin. Which events to include is determined by the focus of the process discovery project. The focus of the project can be on a certain part of the process. This means that preferably (more detailed) events of this part should be included in the event log. While events of other parts of the process could be omitted.

It is important to explicitly choose the level of detail of events. In general the entire event log should be at a consistent level of detail. During process discovery all events are treated equal. A uniform level of detail is therefore important for correct conclusions to be drawn in the analysis phase.

In many applications, the events in the event log are at a different level of granularity than the activities relevant for end users. Some systems produce low - level events that are too detailed to be presented to stakeholders interested in managing or improving the process. Fortunately, there are several approaches to preprocess low - level event logs[8].

Challenge 5: event correlation

Events in an event log are grouped per case. This simple requirement can be quite challenging as it requires *event correlation*, i.e. events need to be related to each other. Consider for example event data scattered over multiple tables or even multiple systems. How to identify events and their corresponding cases? Also consider messages exchanged with other organizations. How to relate responses to the original requests? When designing logging functionality from scratch, it is quite easy to address this problem. However, when dealing with legacy and a variety or interconnected systems, additional efforts are needed to correlate events[9].

Challenge 6: timestamps

Events need to be ordered per case. In principle such ordering does not require timestamps. However, when merging data from different sources, one typically needs to depend on timestamps to sort events (in order of occurrence). This may be problematic because of multiple clocks and delayed recording. For example, in an X-ray machine the different components have local clocks and events are often queueing before being recorded. Therefore, there may be significant differences between the actual time an event takes place and its timestamp in the log. As a result the ordering of events is unreliable, e.g., cause and effect may be reversed. In other applications, timestamps may be too coarse. In fact, many information systems only record a date and not a timestamp. For example, most events in a hospital are recorded in the hospital information system based on a patient id and a date without storing the actual time of the test or visit. As a result, it is impossible to reconstruct the order of events on a given day. One way to address this problem is to assume only a partial ordering of events (i.e. not a total order) and subsequently use dedicated process discovery algorithms for this. Another way to (partially) address the problem is to “guess” the order based on domain knowledge or frequent patterns across days.

4. CASE STUDY

The basic idea of our approach to extract process–related information from a data source is demonstrated by performing a case study. Our case study is related to a standard sales order processing, that is, business process from sales quotation through sales order and delivery to invoicing.

In this process we identified the following activities: Create Sales Quotation, Send Sales Quotation, Sales Order Registration, Approval of the Order, Goods

Issue, Invoice Creation. The extracted data is stored in a PostgreSQL database of a Graffiti.ERP system (from PC GUARD S.A.) with a file size of 315 MB.

Table 1. Activity to Table mapping

Activity	Table	Timestamp attribute
Create Sales Quotation	mzk_oferty_naglowki mzk_oferty_pozycje	data_wyst_oferty
Send Sales Quotation	mzk_oferty_naglowki mzk_oferty_pozycje	data_wyslania_oferty
Sales Order Registration	mzk_zamow_klienta mzk_zamow_klienta_pozycje	data_wystawil czas_wystawil
Approval of the Order	mzk_zamow_klienta mzk_zamow_klienta_pozycje	data_zatwierdzil czas_zatwierdzil
Goods Issue	gm_dokumenty_naglowki gm_dokumenty_pozycje	data_wystaw czas_wystaw
Invoice Creation	spd_faktura_naglowki spd_faktura_pozycje	data_wyst

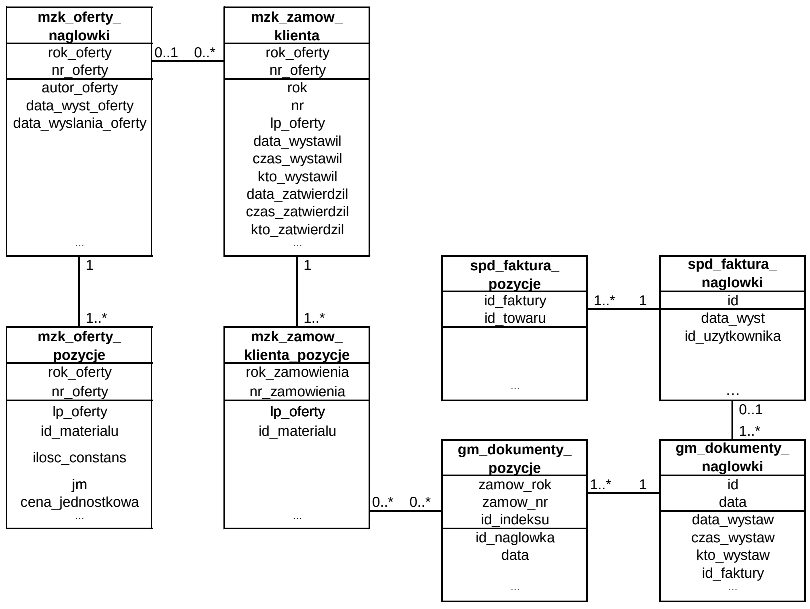

Fig. 4 Sales part of an Graffiti. ERP data model

A first step in determining the relations between executed activities is to identify the base tables in which information about the activities is stored. The base table for an activity is the table where the most important information for that activity is stored. For example, creating a sales quotation produces a new record(s) in the *mzk_ofer-*

ty_naglowki table and in the *mzk_oferty_pozycje* table. The base tables we identified for the activity Create Sales Quotation are thus *mzk_oferty_naglowki* and *mzk_oferty_pozycje*. Table 1 gives a mapping from activities from the our process to their base tables. The source database contains 1379 tables, of which we will use 8. Figure 4 shows a selection of tables and their fields from the data storage.

Table 2. Activity to Table mapping

Activity	Table	Timestamp attribute
Create Sales Quotation	mzk_oferty_naglowki mzk_oferty_pozycje	data_wyst_oferty
Send Sales Quotation	mzk_oferty_naglowki mzk_oferty_pozycje	data_wyslania_oferty
Sales Order Registration	mzk_zamow_klienta mzk_zamow_klienta_pozycje	data_wystawil czas_wystawil
Approval of the Order	mzk_zamow_klienta mzk_zamow_klienta_pozycje	data_zatwierdzil czas_zatwierdzil
Goods Issue	gm_dokumenty_naglowki gm_dokumenty_pozycje	data_wystaw czas_wystaw
Invoice Creation	spd_faktura_naglowki spd_faktura_pozycje	data_wyst

The size of the tables (the number of records in each table) we use to discover of executed process is shown in Tab. 2. The source database contains 1379 tables, of which we will use 8. Fig. 4 shows a selection of tables and their fields from the data storage.

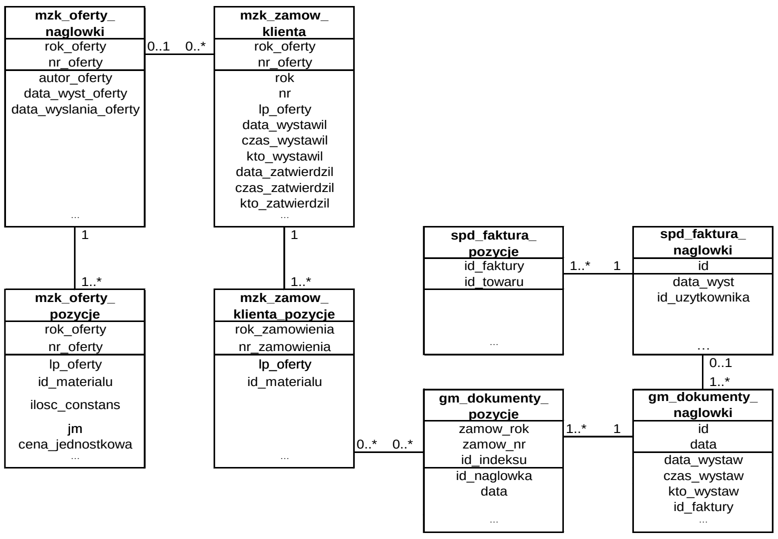

Fig. 4 Sales part of an Graffiti.ERP data model

The size of the tables (the number of records in each table) we use to discover of executed process is shown in Table 2.

Table 2 Number of Records in Sales Quotation to Invoicing Tables

Table	# Records
mzk_oferty_naglowki	2 125
mzk_oferty_pozycje	15 251
mzk_zamow_klienta	68 155
mzk_zamow_klienta_pozycje	210 651
gm_dokumenty_naglowki	368 758
gm_dokumenty_pozycje	1 317 272
spd_faktura_naglowki	146 784
spd_faktura_pozycje	976 337

The next challenge is to find a trace ID that is the common denominator of the selected activities.

The Graffiti.ERP system is object centered information system, i.e. it stores documents[10] that are related to activities in the sales process[11]. For example, the creation of a sales order results in the creation of a sales document, shipping the goods will result in a delivery document, preparing the invoice gives a payment document, etc. Furthermore, one sales document can relate to one or more deliveries, and every delivery can relate to one or more payment documents, etc. In order to deliver executed process, the transformation from object centered to process centered approach is a step that needs to be performed in the data extraction phase. The most important step in the transformation from object centered to process centered is the definition of an appropriate trace ID. An appropriate trace ID is a trace ID that generates an event log with the right level of detail and the right characteristics to realize the process discovery project goals.

In the extraction we used the order line as our trace instance. Per trace we include particular information elements about the order line which are stored in the all eight tables.

The table *mzk_zamow_klienta_pozycje* stores order line information. Each order line is related to an order as stored in table *mzk_zamow_klienta*. Each order has a unique number stored in the *rok* and *nr* fields of the *mzk_zamow_klienta* table (the *rok_zamowienia* and *nr_zamowienia* fields of the *mzk_zamow_klienta_pozycje* table refers to the order in the *mzk_zamow_klienta* table it belons to). Each order line is identifiable by a combination of the order identifier and a separate order line identifier stored in the *id_materialu* field of the *mzk_zamow_klienta_pozycje* table. Other attributes included in the *mzk_zamow_klienta_pozycje* table for order lines are the unit and the value of the order line.

The *mzk_oferty_naglowki*, *mzk_oferty_pozycje*, *gm_dokumenty_naglowki*, *gm_dokumenty_pozycje*, *spd_faktura_naglowki* and *spd_faktura_pozycje* tables store the history of sales documents. These tables contain information of the creating a sales quotation, handling of goods and invoices related to the orders.

As a result, a trace ID is a concatenation of the sales order number (header level) with the order line identifier (item level).

The resulting event log contains 6 different activities, containing 104 381 events spread over 15 885 cases.

In addition to a list of all cases in the event log, we also get access to the variants in executed process. A variant is a specific sequence of activities. We can see it as one path from the beginning to the very end of the process. In the process map, an overview of the process flow between activities is shown for all cases together. A variant is then one "run" through this process from the start to the stop symbol, where also loops are unfolded.

Usually, a large portion of cases in data set is following just a few variants. For example, in the our process the top two most frequent variants[12] cover the process flows of ca. 69% of all cases, although there are 132 different variants in total.

The core functionality of process mining is the automated discovery of process maps by interpreting the sequences of activities in resulting event log.

The process model discovered using the Fuzzy miner (Günther, 2007) is shown in Figure 5. This process model displays each activity in the event log (see our six activities in the main flow of activities). The arrows between activities indicate the order in which they are executed for traces in the event log. Figure 5 presents the model where 57% of the cases are included Mining the model on 100% of the cases results in a 'spaghetti' like model (van der Aalst, 2011, p. 301)[13].

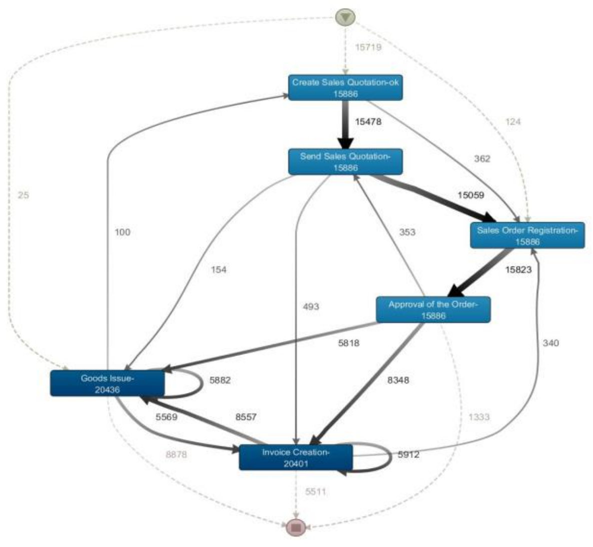

Fig. 5 Exploring 57% of the Cases

5. CONCLUSION

Within this paper we analyzed the extraction of event logs from non-event-based data sources.

We propose to use process mining techniques to truly "close" the BPM life-cycle. The contribution we made was a discussion on important aspects to consider when defining a extraction of process-related information to an event log format. These aspects should be taken into account in every process discovery project.

To demonstrate the applicability of our proposed solution a case study have been performed on data from ERP system.

We believe that we have shown and supported a generic approach to extract event related data from a data source without the need to program. Main improvement would be to investigate and implement a way to visualize the extraction definition. It allows a extraction definition to be printed and discussed with business experts in a clear and concise way.

REFERENCES

Bose R.P.J.C. & van der Aalst W.M.P., (2009), "Abstractions in Process Mining: A Taxonomy of Patterns", U. Dayal, J. Eder, J. Koehler, and H. Reijers, editors, Business Process Management (BPM 2009), volume 5701 of Lecture Notes in Computer Science, pages 159 - 175. Springer, Berlin.

Ferreira D. R. & Gillblad D., (2009), "Discovering Process Models from Unlabelled Event Logs", U. Dayal, J. Eder, J. Koehler, and H. Reijers, editors, Business Process Management (BPM 2009), volume 5701 of Lecture Notes in Computer Science, pages 143 – 158, Springer, Berlin.

Günther C. W., (2007), "Fuzzy Mining: Adaptive Process Simplification Based on Multi-Perspective Metrics, G. Alonso, P. Dadam and M. Rosemann, editors, International Conference on Business Process Management (BPM 2007), volume 4714 of Lecture Notes in Computer Science, pages 328 – 343, Springer, Berlin.

Process Mining Manifesto, (2012), F. Daniel et al. (Eds.): BPM 2011 Workshops, Part I, LNBIP 99, pp. 169–194, Springer, Berlin.

van der Aalst W. M. P., (2004), "Business process management demystified: a tutorial on models, systems and standards for workflow management," Lectures on Concurrency and Petri Nets, J. Desel, W. Reisig, and G. Rozenberg, Eds., vol. 3098 of Lecture Notes in Computer Science, pp. 1–65, Springer, Berlin.

van der Aalst W.M.P., (2011), Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Berlin.

van der Aalst W. M. P., ter Hofstede A. H. M. & Weske M., (2003), "Business process management: a survey," Proceedings of the International Conference on Business Process Management (BPM'03), W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, Eds., vol. 2678 of Lecture Notes in Computer Science, pp.1–12, Springer, Berlin.

Weske M., (2007), Business Process Management: Concepts, Languages, Architectures, Springer, Berlin.

BIOGRAPHICAL NOTES

Wojciech Fliegner is a Professor at the Poznan University of economics. His major areas of academic interest include decision theory, computatonal science, modelling and simulation, software engineering. He has been working as an ERP and the business processes consultant in industry and finance.

[1] The notion of a process model is foundational for BPM. A process model aims to capture the different ways in which a case (i.e., process instance) can be handled. A plethora of notations exists to model operational business processes (e.g., Petri nets, BPMN, UML, and EPCs). These notations have in common that processes are described in terms of activities (i.e., a well-defined step in the process) and possibly subprocesses. The ordering of these activities is modeled by describing causal dependencies. Moreover, the process model may also describe temporal properties, specify the creation and use of data, for example, to model decisions, and stipulate the way that resources interact with the process (e.g., roles, allocation rules, and priorities).

[2] Process mining in our approach is the art of extracting non-trivial and useful information about process from a recording of its execution.

[3] Each of three well-known process discovery algorithms: the α-algorithm, the heuristics miner and the ILP-Miner produces results that are in, or can be translated to, the Petri net modeling notation.

[4] One of the challenges of process discovery is to balance between "overfitting" (the model is to specific and only allows "accidental behavior" observed) and "underfitting" (the model is too general and allows for behavior unrelated to the behavior observed).

[5] As such, events don't have a duration.

[6] This computer language is developed for working with databases and is the most widely used language for relational databases. Using SQL as the language within our extraction has many benefits. First of all it is not a programming language but a query language and therefore has a much simpler syntax. Furthermore, it is possible to split up the elements of a complete SQL query in smaller parts. This further simplifies the creation of the entire extraction definition.

[7] The total number of queries is therefore equal to the total number of event definitions plus 2 (one log and one trace definition).

[8] For this purpose the Architecture of Information Systems research group at Eindhoven University of Technology specified the MXML event log format. The successor of MXML is XES, the format adopted by the IEEE Task Force on Process Mining.

[9] For example, in (Bose & van der Aalst, 2009) it is shown that frequently appearing low - level patterns can be abstracted into events representing activities.

[10] See (Ferreira & Gillblad, 2009) for an example of an approach to correlate events without any a priori information.

[11] Sales documents in Graffiti.ERP system consist of different levels of detail:

- Header: Data in the header of a sales document are applicable to the entire document. This includes, e.g., customer-related data,

Items: Each item of a sales document contains its own data. This includes, for example, material data and order quantities. Each sales document can contain multiple items, while each individual item can be processed differently. Examples are material items, service items, free- of- charge items or text items.

[12] An example of a process centered information system is a workflow system that supports a customer service process. Every service request is a case and all actions that are performed to answer the request are logged as activities related to the case.

[13] Variant 1: Create Sales Quotation → Send Sales Quotation →, Sales Order Registration → Approval of the Order → Goods Issue → Invoice Creation.
Variant 2:Create Sales Quotation → Send Sales Quotation →, Sales Order Registration → Approval of the Order → Invoice Creation→ Goods Issue.

[14] Mining the model on 100% of the cases results in a 'spaghetti' like model (van der Aalst, 2011, p. 301).